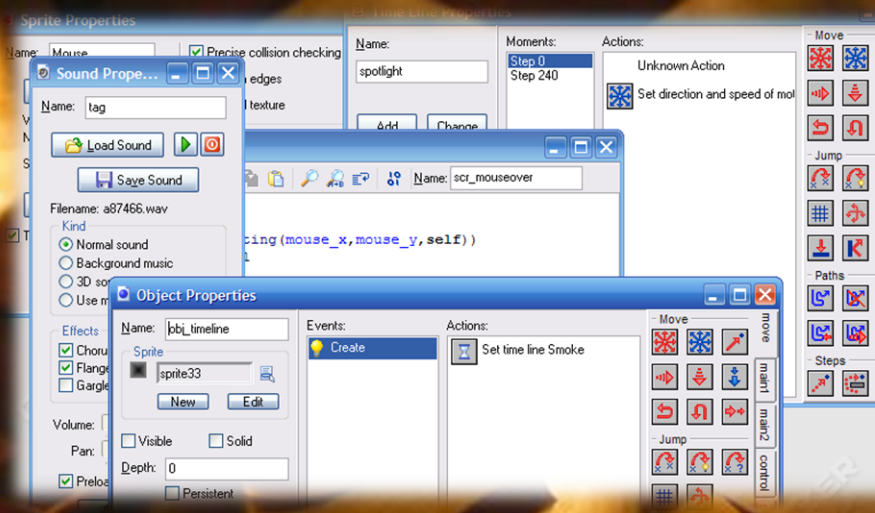
A GMking.org Publication

MarkUp

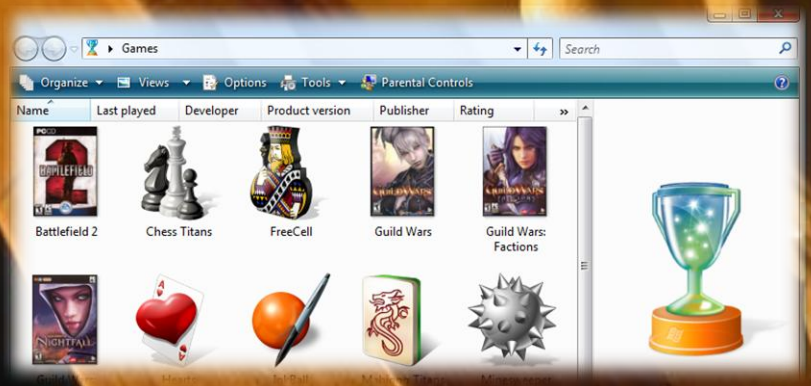
April, 2007

Game Maker 7 is OUT!



What you need to know (pg. 6)

Plus...



Integrate with Vista's game explorer (pg. 17)

Play games of years gone by (pg. 2)

Markup is an open publication made possible by the contributions of people like you; please visit markup.gmking.org for information on how to contribute. Thank you for your support!

Editor's Desk

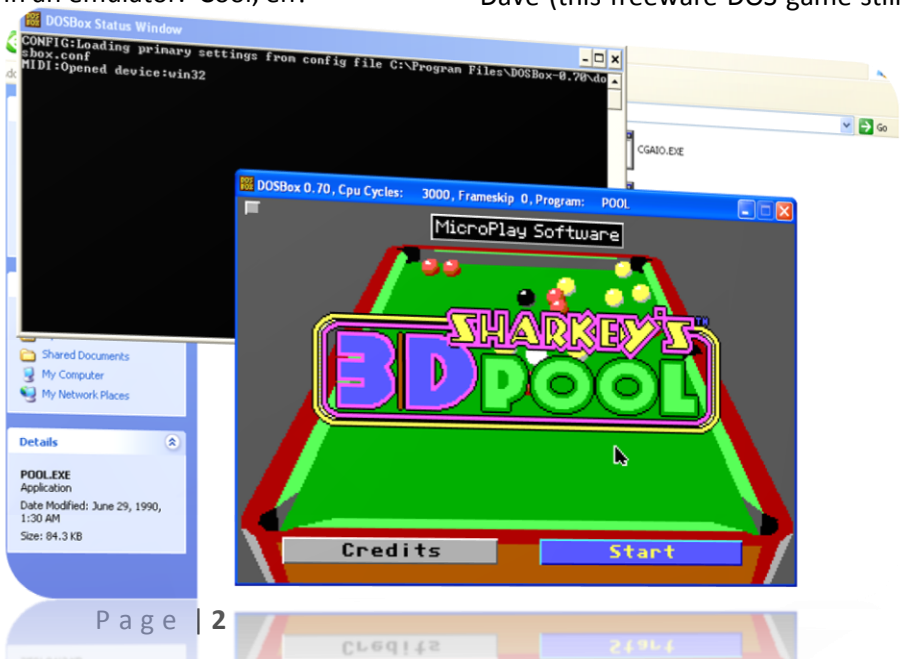
Games of Years Gone By

Remember the days of DOS? You'd get a new game (on a floppy) from a friend and Ooo and Ahh over the simple physics that could keep you occupied for hours?

Fast forward around 15 years and we have games that require two CDs or a DVD to install (over a thousand floppies) and we still Ooo and Ahh, and enjoy them just as much as we did when we first put in a copy of SimCity for DOS, or Sharky's 3D Pool.

If you're like me you have boxes full of old games and apps on floppies for DOS, along with some of those "800+ Games" CDs that came out for Windows, slowly fading away. Why?

A lot of those games can still be run, using emulation software like DOSBox (dosbox.sf.net; pictured below) for your DOS and Win 3.1 (you remember 3.1 right?) games. If you still have a legal copy of Windows 3.1 (or in my case, 3.11 for workgroups) you can get a copy of the free VMWare Server (vmware.com) and install the actual Win 3.1 in an emulator. Cool, eh?



And for those of you with some old game system cartridges around, you can download copies of the games you own (it's only legal if you already owned the original cartridge) that will run in emulators like mame (mame.net) at theoldcomputer.com.

Balloon Challenge

Digging up some of those games can bring back some of the fun gaming possibilities of some of the great games of yesteryear. Some of my personal favorites I've resurrected are Sharky's 3D pool shark (which I picked up a copy of a few years back in a dollar bin at the local geek shop), Balloon Challenge (got that in a 500 game pack, it's a shareware game, so you can still pick it up at xrl.us/bchallenge), Dangerous Dave (this freeware DOS game still runs perfectly on WinXP, xrl.us/ddave) and In Search of Dr. Riptide (also available as Shareware xrl.us/driptide).

Which brings up another good point, there are lots of great shareware games for DOS still floating around. Back then, the demos went just a movie of screenshots; they were hours-to-days of fun games. You'll find a lot of them available on DOS archive sites such as dosgamesonline.com.

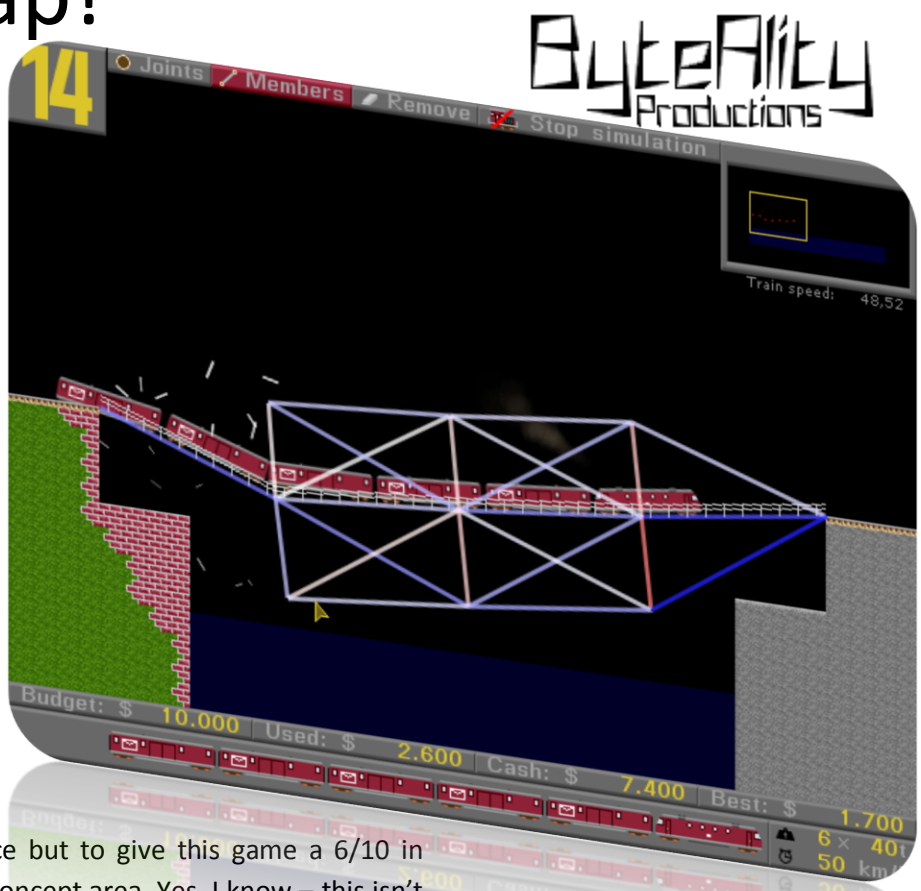
Switching from the past of gaming, to the future; open source games are becoming more popular, more advances and much better. Here are just some of the great games I've run into recently (all of these run in Windows, most will also run on Linux & Macs):

Scorched3D . sf.net/projects/scorched3d
Bacteria ..... bacteria.simondonkers.com
Planet Penguin Racer ..... xrl.us/PPRacer
Enigma..... www.nongnu.org/enigma
Frozen Bubble..... frozen-bubble.org

See you next month!
Robin Monks, mozillaman.uni.cc

In This Issue...

Bridging the Gap!



I haven't been active on the Game Maker Community for a while, I don't check topics and creations as much, and I'm probably not as up-to-date as I used to be. But I do check my PM daily, and that's when I knew a creation of mine was in the cage match against this wonderful game: Bridging the Gap!

The game was created by Erik Leppen and so far: I'm loving it! There are some very good points about this game that I will surely discuss, and other bad points which I'm also going to talk about.

Concept and Originality

If I just look at the game and isolate it, I have to say it has an excellent concept – and it is fun! But people like me who used to spend a lot of time in Download.com trying to find a nice game to play would find this game very familiar.

When you look at the screenshot of the actual game, and compare with the screenshot of "bridge builder game" on the next page, you will notice the resemblance -- its striking!

Now I would've liked to see a "completely" original game that I could review, but that doesn't make the game – overall – any worse, except for the fact that coming up with the idea itself wasn't the author's task.

For information about "Bridge Building Game", see bridgebuilder-game.com, or for the version of the game I originally used to play, see download.com/3003-2111_4-5862232.html.

For all of the reasons above, I have no choice but to give this game a 6/10 in the concept area. Yes, I know – this isn't particularly good, but keep in mind, it is a great implementation of the game as you will see soon.

Music and Sounds

So I kept saying how good the game was, but I started out with a bad point about the game, and I decided to keep it that way: so here's the next bad thing in the game: Music!

We need music, we need sound effects, and we need something to listen to while we're playing the game! I don't care how fun anyone thinks the game is, because it won't be as fun as it would've been with good music.

So, all I can do is give the game a 3 out of 10 in the music section.

Graphics

The question is: what kind of graphics do we see in this game: well, there's the bridge, there's the ground, there's the actual train, and some graphic effects (which we will cover later). If you ask a graphic designer what s/he'd think of them, you probably won't hear much praising – but from a game play and design perspective: excellent.

The graphics are what I like to call "effort effective", which means that whoever made the graphics put in the amount of effort that would make the game good (in terms of game play), yet "worth the effort". Now sure, more effort – and therefore better graphics – would've made the game even better, but it simply gets to the point where it's simply better to keep them the way they are.

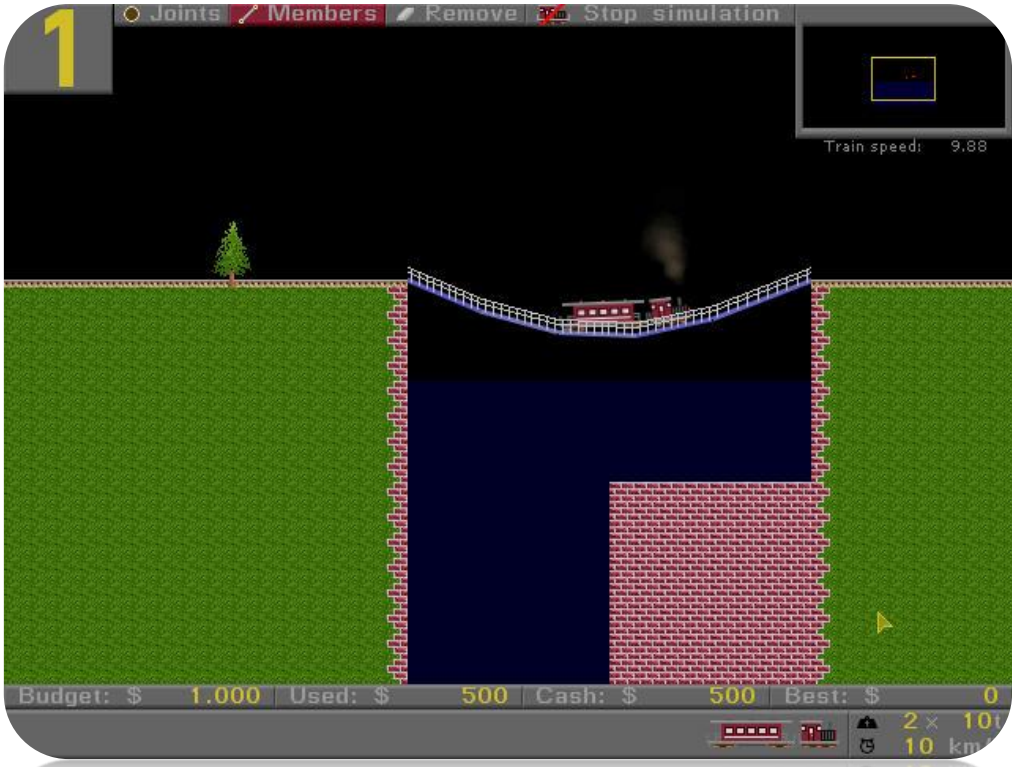
Bridging the Gap! Cont.

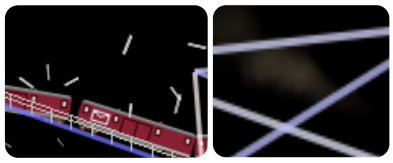
I give the graphics 8/10, they well suit the game, but of course – there's always space for improvement.

Graphics Effects

So there are two types of graphics effects that I have noticed: destruction, and smoke – but I'm sure I might find more if I play it a bit longer.

Graphics Effects are always a plus to have, but not an essential element in game play. When looking at the smoke alone, I like it – the way it is realistic and smooth, and when looking at the destruction effects, I also like it: it fits the rest of the game's graphics and provide this certain "feel" to the game that enriches the experience of the player.





But again, the thing is, these effects are almost exact opposites of each other – either make them all realistic and smooth, or all cartoonish and 'weird'. Both are good, just not together, and for that, the game gets another 8.5/10.

Game Play (Controls)

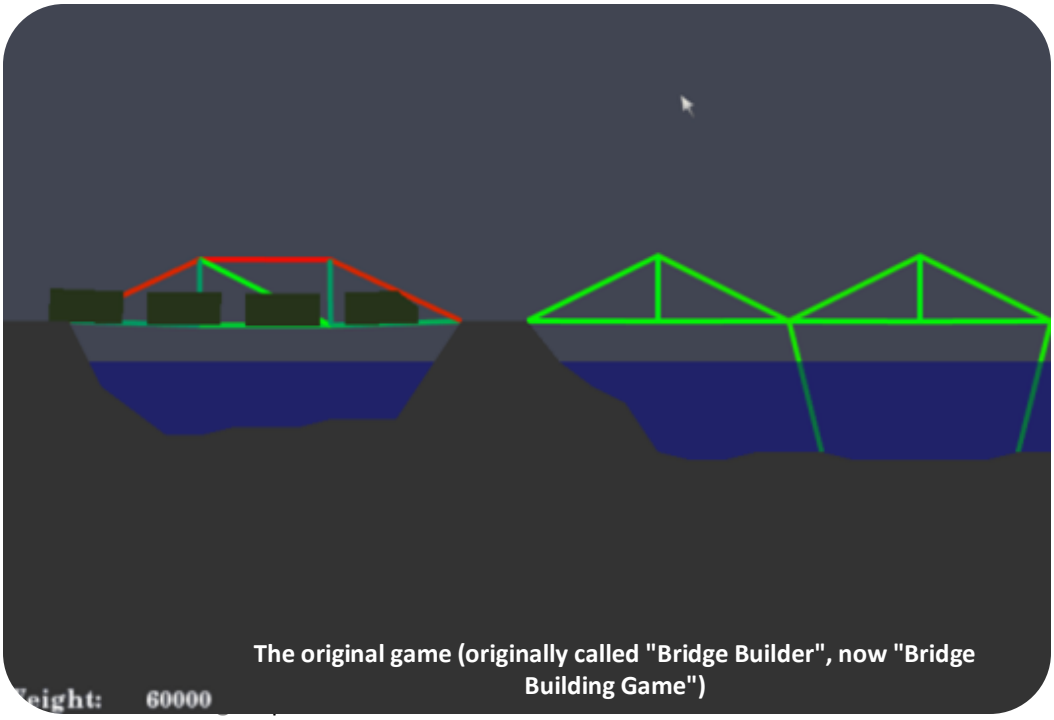
The controls are good, easy to use and understand, and most importantly: simple in every way! I've only played it a couple of times and learnt a few control tricks, such as clicking on a single point and connect all possible bars to it, etc.

The controls themselves are excellent, different controls are accessible from the top bar as well as a quick right-click for those who can't wait, or are simply pros in the game and like to do everything quickly.

I have no other choice but to give the game a full 10/10 in Game controls.

Game Play/Player Experience

This game is fun, fun, fun! Most people would love, whether or not they are interested in physics, building, bridges, cars, trains, computers, whatever – it's a must-try!



The original game (originally called "Bridge Builder", now "Bridge Building Game")

Bridging the Gap! Cont.

There is one disadvantage in the game: when testing the bridge by allowing a train to cross it, the train moved too slowly, and I couldn't find a way to speed the train up, or even skip the whole part – I really think such feature should be available.

The game gets 9.5/10 for player experience.

Versatility

"Versatility", in each game, is measured in different ways. Sometimes it's in the level design, in other cases it is in the atmosphere of the game, etc. Here however, versatility counts in the variety of levels the game has – and here's what I've heard: it has 50 levels! Now that should keep you free for a while, eh?

I must give Erik's game a 10 in versatility, because the effort itself was amazing – and in each new level: a new, different challenge!

Realism

So how realistic is this game? Will such bridges hold that weight in real life? Well, according to what I've seen: it should. The game is based on a realistic and awesome (in my point of view) physics engine. It's all done quite well, stress is analyzed, and according to that the bridge breaks or holds.

Of course, don't take my word as an answer; in real life, there are many variables and such things are much more difficult and complex to calculate, so I'll just say that the game physics is "sufficient". And by saying sufficient, I mean 9/10.

Programming and Bugs

Well, the programming is definitely amazing, I must say. I have personally found no bugs and I've enjoyed playing it, but I've checked the topic and found a single bug. I tried to track it and see whether or not was it resolved, but I didn't find out much.

So for programming, the game gets 9/10 – because it is one good "replication" (or even enhancement) for the original Bridge Building Game.

Overall

When adding up all the scores, I found that the game's total score is 7/10. I personally hoped the game would get a higher score, but there's nothing much that could be done.

Again I must say the game is incredibly fun, and a must-try. Such a game tells us one thing about game development: a game is never limited by its tool or language; it is only limited by the skill, effort, and creativity of its author, and what a creative author he is. Bravo.

Eyas Sharaiha

TUNTIS' RANT

PHP

Hi, my name is "tuntis", and I've been a regular user of the GMKing.org services. About 4 months ago from the moment of writing, I decided to pick up learning PHP as a programming language.

I, currently, have created a lot with it, and know all the basics, and even run an online game, which I decided to pick up after it's original creator went on to another project of his own. I've extended it a bit, and continue to run it to the current day.

However, there have been some things I haven't liked about PHP, namely the following:

Error Reporting

Error reporting could use much more beginner-friendly reports (although one does get used to the most common ones), but can really confuse new programmers sometimes.

Image Functions

Much isn't needed to describe my problem with them: they are just a "bit" too hard to use. Allocating colors to use, etc can be difficult.

Otherwise, PHP has been quite a nice "adventure", with the exception of the lack of...

Proper Online Documentation

The PHP manual is a great resource, but its examples are usually too incomplete and/or undocumented.

There are not enough tutorial sites and looking at some other application's source code doesn't really help.

tuntis - http://www.tuntis.net

GAME DEVELOPER IDES

Game Maker 7.0

Introduction

Those interested about Game Maker have gone crazy in late July, when Mark announced that the next version of Game Maker will not be 6.2, as previously announced, but instead *7.0*. Well, with such a change in the version numbering, you would have expected much more features and something "completely new", some people say that is exactly what Game Maker 7.0 is, while others remain unsatisfied by the lack of sufficient new features to give the 7.0 number to this version of Game Maker.

However, whether people are satisfied by the version numbering of Game Maker or not doesn't change the software within. I think it is silly that some people wouldn't like Game Maker 7.0 just because of its version number – it could be version 11 or 2.3 and still be as good! So, whether this version of Game Maker lived up to its number or came as a disappointment, the review won't change – the only question that truly matter is: is Game Maker 7.0 a good addition over Game Maker 6.1, and is it worth to upgrade?

This review will go through all of the new features of Game Maker 7.0 in incredible detail, and cover every change that has occurred in it, so that I could come up with a result to answer the question asked above.

The General Interface

The first thing we notice when looking at the general Game Maker interface is a new item the left pane called "Extension Packages", and two new toolbar items on the top: "Create a stand-alone executable for your game" and "Publish your finished game on the website". This section will not go through any of these features in detail, but will instead be concerned about their integration with the Interface.

So, the Game Maker interface is the same. Not much is changed; we're still used to it, just some extra icons and a nice little YoYo Games image on the bottom of the left resource pane. However, a point worth noting is that that small link could be removed from the interface from the options menu.

The changes to the interface are incredibly minor, and are done in such a way that wouldn't affect those who are used to previous versions of Game Maker. Buttons are added to the same interface, the same way other buttons were added before. In terms of consistency, it is incredibly consistent – and that is the number one most important thing in the UI of any software.

Incompatibilities

Game Maker 7.0 has one incredibly positive point: it has too few incompatibilities. So what? The file extension has changes, big deal. However, the real problem is that room transitions are incompatible with previous versions. Mark Overmars describes this problem as a "minor incompatibility with room transitions", but at the end of the day, you'll probably end up having to rework some of the transitions you have in your game, *if* it had lots of them.

Major Features

Those who are disappointed by the version number are probably upset about this section! Yes, it *is* true; Game Maker 7.0 has only a few "major" features and additions. But, this hopefully shouldn't stop it from being a good, solid addition to the Game Maker line.

The Extension Mechanism

The Extension Mechanism has an excellent concept *and* implementation! The general of concept of the extension mechanism is that any user could be capable of creating "Extension packages", and integrate them at a deeper level that before. Extension packages could be one of three: script (GML), DLL, and a Library. Now, what differentiates extension packages between regular scripts, DLLs, or Libraries is that Extension packages are integrated differently; they become actual scripts which the user might use, and it would seem like they are *part* of Game Maker itself. They have their own help, accessible via Game Maker, they are mentioned in the function list, like all other Game Maker functions are, they could have their own constants, etc. They are basically an actual addition to Game Maker itself, except it isn't written by Mark, and it's either a GML script, a DLL, or an Action Library.

What confuses me is the action library

Game Maker 7.0 Cont.

part; libraries are already being integrated into Game Maker – at a different level, true, but I think that was enough. Integrating them in this way as well would cause confusion: where is this library? Is it in an extension package? Or maybe it is in the lib folder? The thing is, they now have their own help – and that's the positive thing.

What occurred to me was that Mark could move all libraries to Extension Packages, but then another problem arises: compatibility. I guess it is difficult to figure something out, but for the extension packages (or at least the library part) to be better and probably less confusing, Mark should've thought of something.

Let's examine the process of adding/removing an installed extension to the Game. We click on Extension Packages, we click on the available packages and choose the package we want, an arrow appears, we click it, boom! Here it is! To remove an extension, we click on Used Packages, click the package we want to remove, we click on the arrow that appears, and it's gone! To summarize, it's easy, smooth, and a very enjoyable process overall! I like it, there's nothing wrong with it, and I personally don't think it could've been done in a better way – this is what it needs to be.

File Inclusion

A rather controversial move Mark did in Game Maker 6.0 was the removal of the Data File resource type, in favor of a much simpler and more limited file inclusion mechanism. The reason was that Mark wanted to keep Game Maker simpler and easier, and now, he's returning some of the features we saw in the Data File resource to the inclusion mechanism.

This has resulted in a simple and understandable file inclusion mechanism – Just like Mark wants, and at the same time a fully capable way of storing and handling files – Just like experienced members wanted. It's a win-win!

Windows Vista Compatibility

Previous versions of Game Maker have had a problem running on the final version of Windows Vista. So, in Game Maker 7.0, Mark fixes that problem! Yes, Game Maker now works on Windows Vista! What's the trick? There's an additional 1 MB of runner data attached to every Game Maker 7.0 executable, nice.

So, no big deal; most of us have broadband and don't care about an addition 1 MB of file size, true. But the other problem that Game Maker has is that when it runs on Windows Vista, a User Account Control (UAC) Dialog appears that asks if you want to run this executable, not nice. But of course, you get that kind of dialog for every unsigned, unknown-authored executable, not just Game Maker executables.

Smaller Features

Be afraid not, my faithful readers; for there are many other smaller features in store, some of which are very nice, and others which are useless!

Publishing Games

On the Game Maker part, it is only a simple button that links to a web page, no big deal. But it *is* a big feature, guess why? Mark is finally making it simpler for newbies to publish their games onto the web via the Game Maker website! Bravo!

Splash Screens

Splash screen functionality has been expanded? I doubt many people will be happy about it, but I for one am. I haven't seen many people use splash screens in Game Maker, even though splash screens are considered an important part of the presentation of the game, before it is actually played. Hopefully now with better splash screen functions, we would see more splash screens used in Game Maker in the future.

The Separate Close Event

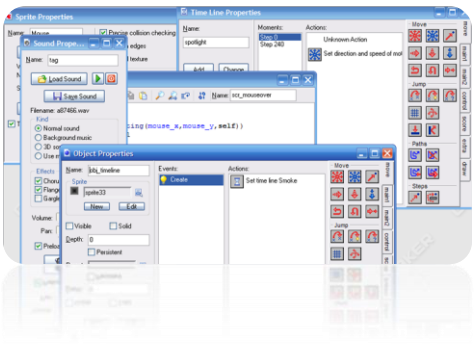
So that's a nice thing, you bet. Having a separate event for the close button is definitely a smart thing, and has already helped me working on a couple of projects that I quickly converter to Game Maker 7 to test it in order to make this review. It's useful, and it doesn't waste space or anything – it's just added to the other events list!

It's the Pro Version!

Instead of having a "Registered" version of Game Maker, now members with a registered copy would notice that their

Game Maker 7.0 Cont.

copy of Game Maker is labeled "Pro". Additionally, the unregistered version is now called the "Lite" version. There is absolutely no underlying change: just the name.



Even Smaller Features

I've said before and I'll say it again: I'm going to cover every single change made in Game Maker 7.0, and I mean it! If I forget about one, then tell me and I'll add it! Here's the list of even smaller and more insignificant features:

Move Actions

The move actions in the "Move Action Library" have had their buttons updated, so when you click on the blue arrows for moving actions, they turn red. Very small change, but nice, and would probably help the user recognize which buttons are being pressed in an easier way.

Showing Loading Bar

Now this is a nice move, the loading bar of the game now appears "*considerably earlier*" than before. This means that you don't actually wait for a few seconds wondering if the game started running when you click run! Good job.

Functions Renamed

The functions in all official Game Maker action libraries have been changed to be compatible with the book. This isn't really a new feature to Game Maker 7.0, as it has been introduced in Game Maker 6.1a, but the thing is: Mark hasn't released that over the internet, so it *is* a new feature for most users, since they don't have the book.

Game Maker now accepts more File Formats

That is truly good news! Functionality for functions concerning splashes screens, sprites, and backgrounds used in GML to load a certain image or video from a file during run time have been enhanced to accept more file formats. Yes! That should have been done since GM 3!

Select All in Image Editor

You are now capable of using the Ctrl+A keyboard shortcut, or the "Select All" command to select all of the contents of the image. Game Maker has always needed that since Game Maker 6, where selection was first introduced.

Centering a Message

The function message_position() has always existed, what's new however is that setting the X and/or Y arguments to -1 centers the message! I've always thought that ability had excited before! Well, apparently it did not, and at least Mark added it now.

What's that little extra button?

Oh yes! It's a print button! Now, on the script and code editors, you will be capable of seeing a print button that prints the code. I understand how some people might find this function useful, but I am not too excited about it. But I don't mind having an extra button that I don't use.

Save and Load Debug Info

So the debug info forms now have new save and load buttons added to them. I personally have never found the whole debugging to be that useful, but I admit to have used it on multiple occasions – but save and load? I'm not sure how it would be useful, but apparently Mark does.

View Events: Outside View and Intersect Boundary

I'm serious: I've needed those events for my game! These are very useful events, especially for those games with huge rooms and enormous amounts of instances.

Showing Errors

So that's a nice little feature: errors are now shown in a text area, nice! I can copy, that's one thing, but also: huge errors (as in errors where the line of code they are in has too much doe) scroll over and do not result in a huge window that blocks the entire screen!

Getting, Setting, and Randomizing the Seed for Random Functions

What? I read the help section about that twice before I got it. Basically, it allows you to set a seed used to generate random numbers. Of course,

Game Maker 7.0 Cont.

random number generation doesn't solely depend on this value.

Navigating through a Sprite's Sub-Images

So that's nice: you can now navigate to the next and previous sub-image of a sprite. This will speed up editing a lot, won't it? Nice feature, and bugs reported in betas have been solved, so don't expect seeing errors or having problems when using this feature. I think it would be particularly useful when someone is working on manually animating a sprite. That way the spriter could move directly between the two frames and manipulate each frame accordingly.

The 'globalvar' Declaration

Many people use the "var" to declare variables, well guess what? You can now use "globalvar" to declare global variables! What's the use, you ask? Well, after declaring a global variable in such a way, you are not required to write "global." before the variable name.

F9 Function Key Takes Screenshots

Pressing the F9 function key now takes a screenshot of the game. This mini-feature could be disabled via the Global Game Settings, so don't worry if you don't want people to use your sprites or take screenshots of your game for any other reason.

Draw a line with a width!

That is a function that we have indeed missed in Game Maker 6. The pen_size variable (or whatever it was called) provided a very similar effect in Game Maker 5, but had been removed from Game Maker 6, since it used Direct3D. Well, now Game Maker has a function that would mimic part of that old function, yay!

IS IT DAMN SUCCESSFUL?

Now we can finally know! Functions for starting and stopping d3d_ now finally return whether or not they are successful. Thank you!

Setting the Cursor Sprite

Apparently we have a variable for that: cursor_sprite. From what I've heard it had been around for a while, but Mark only documented it now.

We Need More Arguments!

And we have more arguments! DLL Functions now support up to 16 arguments, cool indeed!

Major Fixes

A major fix is the encryption of scripts via runtime, and other protections to the executable file of Game Maker games. These fixes make it much harder for people to decompile a game and to view or use the code by accessing it via the RAM on runtime. This is an excellent security fix and should save great games' sources for being viewed by unauthorized people.

Please note that I am not in any way here talking about the .gmk encryption, I am talking about encryption to the executable and scripts themselves when they are being run.

Bugs Corrected

Mark has corrected a handful of bugs and annoyances in Game Maker 7.0, and no: I'm not talking about bugs in the beta itself, I'm talking about bugs in the previous versions of Game Maker. We will now examine these corrected bugs and the impact those fixes will have on the overall game making experience.

Corrupt Files

Here's a cool new feature in Game Maker 7.0: Game Maker is now more capable of reading Game Maker Source files, it doesn't tell you it's corrupt if it still is readable for the most part. Mark claims that most Game Maker files that GM labeled as corrupt will now be readable by Game Maker 7.0! Now that's nice!

Corrected Limited Real Precision Bug – I know the time now!

Game Maker 7.0 has finally solved a bug that existed at least from Game Maker 6.0, where basically real numbers with a certain precision weren't handled correctly. Since date and time in Game Maker are also expressed by real values, getting the date and time wasn't always correct; sometimes it'd tell you it is January 24 when in fact it is January 25! I'm glad this is over, because I had huge date and time problems in one of my projects

Corrected Timeline Bug

A bug in timelines has finally been

Game Maker 7.0 Cont.

corrected! The bug occurred when content is duplicated to an earlier moment in the timeline.

No more Syntax Errors for Bitwise Assignments

Assignments such as =, +=, etc. have always worked properly; because they are the most used. But finally: now, also bitwise assignments work properly! What are bitwise assignments? They are those assignments using bitwise operators, such as &, |, ^, <<, and >>. So now, using &= |=, >>=, etc. should give out no errors, supposedly.

Show Message actions keep box centered

All show message actions keep the show_message function itself centered in the screen (unless the position itself is changed). This is important for providing users with a better experience when playing your Game, even though it's probably not that big of a deal.

Explosions work, Even at cold weather!

I've never knew that, but apparently using explosions with the snow effects at the same time caused problems! Good news: it is now fixed and they both work properly!

Don't Worry About Decompiling!

So in Game Maker 6, we've all heard about Mark's huge efforts to stop decompiling from occurring, and it worked for a while. Though decompiling itself didn't occur for a long time, what people discovered is that they were capable of extracting information from Games at run time. There were some other flaws too, have lead to a successful decompiling attempt. But: Mark fixed that now, thankfully. So we shouldn't be worrying about having our creations subject to decompiling, for the short term at least. It is in general recommended, however, that we keep things like passwords away from our Games' executables.

Less Annoying Debug Forms

Debug forms are now less annoying, Mark says. Debug forms no longer keep jumping to the top position.

No More Multiple Fonts, ALWAYS Color Coding!

So, I bet most of you have had this problem. Copying a piece of code from somewhere like a webpage where it has been formatted results in having it pasted in that same formatting, and no color coding is applied. That has been a very annoying bug in Game Maker, and Mark has finally fixed it in the 7.0 release!

Backgrounds keep their settings

A rather annoying bug was that backgrounds would lose all of their settings when you change the background image. Luckily, it has been fixed in GM7.

Rectangles are now accurately drawn

So, now there is now no difference between the size of a solid rectangle and a border rectangle. Smart.

Empty Window Shown Before Create Event

Were you ever annoyed that when showing messages on Create Event, you'll actually need to move them to a later event (or use redraw functions)? Well now, the window is shown earlier, so from moment the create event is executed, the window is there.

I have my concerns on this "fix", as probably many people were satisfied that the window weren't shown before. There is no way to predict whether or not will this fix backfire, except for waiting.

Median Now works properly

I don't know about the rest of the people, but I personally use the median() function a lot. For those who do not know, the median function calculates the average of a number of values entered, and is really useful in numerous games. Apparently, it had a problem, but it has been fixed in this version.

Resizing Grid: No more crashes

I haven't noticed that bug to be honest, but apparently, Game Maker crashes sometimes when grids are being resized. Mark has finally solved that issue now, but I'm still confused – because I've never saw it.

Grid Data Structures Now Work Well With Strings

The "grid" data structure previously had

Game Maker 7.0 Cont.

problems with storing string values; this has been fixed in Game Maker 7.0.

Sprite Preview Issues – Fixed

The sprite preview pane sometimes showed sprites as being transparent even though they have been set to non-transparent. This issue has now been fixed: all sprites preview correctly.

Collision Checking works with Scaled Instances

If an instance of an object has a scaled sprite, it won't stop you from having proper collision checking in Game Maker 7.0! That's indeed good news.

Collision_ Functions with non-integer Coordinates work properly

All of the collision_ functions that check for collisions between lines, rectangles, circles, etc. and objects now work as expected, even if their coordinates are non-integers.

Opening a Non-Existent Binary File Creates it

As with the rest of file handling functions, the open function for binary files file_bin_open() now creates the file to be opened if it doesn't exist. It's surprising such function didn't exist before.

Setting Object Parent Via GML now has Checks

So, the object setting function of GML now has addition checks to see if any cycles were made in the object's parents. That is incredibly good news, from my point of view.

Bounce Bounces Correctly!

So that's good news: if you have diagonal surfaces, and use the bounce action on collision, the object will now realistically bounce off it, and it wouldn't treat it like horizontal and vertical surfaces.

Is there something wrong with Game Maker 7.0?

You might ask: is there something wrong with Game Maker 7.0? And the answer would be: definitely. Game Maker 7.0 is by far not perfect when we judge it a separate piece of software. However, when looking at it at a simple upgrade, it has no bad points *compared to Game Maker 6.1* (except for the .gmk encryption, we're getting to that later).

So, this means that if you want to use Game Maker as a regular game development tool, you definitely should get Game Maker7.0 –it is *that simple:* Game Maker 7.0 isn't worse than Game Maker 6.1, as a matter of fact: it's better. The other question is: is it what we had expected? And the answer to that could be different from person to person; I'll tell you my point of view after a couple of sections.

The .GMK Encryption

The ".gmk encryption" is a new change in Game Maker introduced by Mark Overmars. It has been highly controversial in the Game Maker community, and many have strongly opposed it, trying to let Mark change his mind about it.

About the .gmk encryption, it refers to the fact that Game Maker 7.0 source files are now encrypted, and their content cannot be read. The encryption isn't a security issue, since the file being encrypted isn't the executable, but instead the source itself!

Here's the thing, Mark never lied: the .gmk encryption *is* aimed to stop projects that access Game Maker source files like G-Java. It is Mark's belief that by stopping projects like G-Java, it is actually in Game Maker's best interest, and he means it.

Users were upset because they thought what Mark was sort of a betrayal to the Game Maker members who have worked hard to provide such tools for Game Maker. They also believe that Mark hasn't gave much thought about incorporating that "feature", and have said that it would do more harm than good to Game Maker, since those making these tools could decide to make their own Game Development toolkits that rival against Game Maker.

In my point of view, however, Mark's probably more right. Converters would create lots of benefit for Game Maker, true. But the important point is, they would create much confusion to the beginner users, it might as well hurt Game Maker's reputation when features in the converters themselves do not work – and that was what concerned Mark.

GAME DEVELOPER IDES

Game Maker 7.0 Cont.

Still though, it is a shame to see such projects fade away, and I would truly hope to still see those years later (on Game Maker 6), to provide "ports" to Game Maker in other languages and systems.

Should I get Game Maker 7.0?

Almost definitely yes! Game Maker 7.0 provides more compatibility, more fixes, and more functionality, the only reason you might stick to using Game Maker 6.1 is to take advantage of the tools like G-Java, which are no longer allowed to work on Game Maker 7.0. Even so, you might consider running both versions of Game Maker simultaneously on your machine, so you could both take advantage of Game Maker 7's new features and the lack of source code encryption Game Maker 6.1 has, if you're considering converting to Java or other languages.

Final Remarks

Game Maker 7 is definitely better than Game Maker 6.x, and is definitely worth to upgrade for a few reasons: it's free and easy to upgrade, so why not get a better version. However, I must really look at Game Maker 7.0 from another perspective and ask myself: is it a disappointing release?

It has few "big features", yes, but it isn't a disappointment –at least for me; because every one of those "small features" would truly make a difference to a Game developer. It *would be* a disappointment however, if we expected the changes to be as significant as the 5.3A->6.0 differences. THEN, it would have been truly disappointing.

Good job, Mark Overmars; this is truly a great tool, and **this** is truly a great release. Some of us would have expected more, but I'm pretty sure we're getting more in the future.

Eyas Sharaiha

TUTORIAL

Game Maker Tutorial

Getting the number of blue pixels on the screen in Game Maker is an excellent example of the return function of Game Maker.

This script makes a loop that checks the color of each pixel on the room. To do this, we check one row at a time, then move on to the next row.

It then returns the number of blue pixels in the current room. Handy if you were making a game that required user interaction with colors, or for a drawing program.

The Example

```
width=room_width;
height=room_height;
current_x=0;
current_y=0;
blue_pixels=0;
repeat(height)
{
    repeat(width)
    {
        current_pixel_color=draw_getpixel(current_x,current_y);
        if current_pixel_color==c_blue
        {
            blue_pixels+=1;
        }
    }
     current_y+=1;
    current_x=0;
}
return blue_pixels;
```

Surfaces

Introduction

Surfaces is a new feature that was added in Game Maker 6.1, it allows the user to use the normal draw functions to draw on a canvas, or a "surface" once, and then, it can be saved to a file, drawn on the screen, go through some manipulations and rotations, or get copied to another surface.

Benefits

Surfaces usually are handy when making a paint system, like MS Paint. You draw a line to the screen, then the line is saved on the surface, and the surface is drawn at the end of each step, which means that even if you draw thousands of lines on the surface, then draw the surface on the screen, the game will still run fast, as the surface is drawn all at once, not each line separately.

Concept

To understand Surfaces more, we have to understand the regular drawing first. To do that, imagine that the main room in any game made with Game Maker is just a simple piece of cardboard, that everything you draw on it is erased after one millisecond. Therefore, if you draw a line once, it would be removed after one millisecond, which means that you need to draw it once after each millisecond is over.

However, Surfaces are just a piece of paper that "flies" above the main room (or the piece of cardboard) and lines drawn on it do not get removed after each millisecond, but they stay drawn on the surface unless you empty (or reset) the surface.

But this is not all what surfaces are about! When you draw a surface, the surface uses the system temporary memory to save its contents, which means it acts as if it is a bitmap file (or something similar) in a file in the Main Memory! So that is why speed is saved when using a surface.

Since we now know what a surface is, I think we can start learning about the code these surfaces use in order to function.

Primary Code

The Basics

```
surface_create(w, h)
```

Creates a new surface with the width "w" and height "h." It then returns the surface "id" which is used in all the other functions. This means that when calling this function, you must set a variable's value to this function, like "_var=surface_create(5,5)"

```
surface_exists(id)
```

Checks whether the surface with the given id exists. Returns true or false (1 or 0, respectively).

```
surface_free(id)
```

Frees the memory that the surface with the given id is using. This must be called when there is a surface you are no longer using in order to save memory.

```
surface_get_width(id)
```

Returns the width of the surface with the given id. The width is what you enter in the "w" field in the "surface_create(w, h)" function.

```
surface_get_height(id)
```

Returns the height of the surface with the given id. The height is what you inter in the "h" field in the "surface_create(w, h)" function.

First Attempt

Ok! So let's try out these functions that we have just learnt in a piece of code, and see what happens! We will create an object called "o_surfaces" and put this code in its **Create Event**.

```
surface=surface_create(50,100);
//Creates the surface
draw_text(5,5,"This text is drawn in the create event.");
/*The code above attempts to draw text in the create event!*/
```

Explanation of First Attempt

The code above creates a surface called "surface" and attempts to use it to draw text in the create event, and therefore take advantage of the surfaces by using a drawing action once to draw on a surface on the screen.

Problem

But after you place the object in the room and run the game, you will notice that no text is drawn! "Why is that?" you wonder!

Reason

The answer is simple! Game Maker doesn't understand automatically when to draw on a certain surface and when you need to draw on the main room, so you need to use certain functions that will switch between drawing on surfaces and the main room, and switch

Surfaces Cont.

between making the drawings on certain surfaces between each other.

Code Explanation will help:

```
surface_set_target(id)
```

This function is the function that tells Game Maker what is the target of the drawing. You need to enter the id of which surface do you want the drawings to be on.

```
surface_reset_target()
```

Resets the drawing target to be on the normal screen again. It is important to use this code immediately when you finish drawing what you need on the surface, as it may mess up the whole game; objects, backgrounds, and other things are drawn on the main screen, making the game draw these things on a surface may corrupt the game.

Second Attempt

```
surface=surface_create(50,100);
//Creates the surface
surface_set_target(surface);
//Sets drawing surface target
draw_text(5,5,"This text is drawn in the create event.");
/*The code above attempts to draw text in the create event!*/
surface_reset_target();/*Resets the drawing target. The drawing target is not set as the main screen in order for the game to continue functioning properly without corruption.*/
```

Problem

Go ahead and test this code, you may think it will work. But it still wouldn't! Do you have any idea why?

Reason

It's simple: the test **is** saved in the surface, and it **is** in the memory, but it **is not drawn** to the game's main screen! It is true that I said you don't have to use the draw functions constantly to draw all the drawings and lines, but however, you do need to draw the surface. So when you draw a 100 lines in the surface, you only need to execute one each step (the surface drawing function), not zero. Below is the code explanation for the surface drawing functions. They have been copied from the help field, and then edited to make the functions clearer.

```
draw_surface(id, x, y)
```

Draws the surface with the given id in the position (x, y) without any colors or alpha blending.

```
draw_surface_stretched(id, x, y, w, h)
```

Draws the surface with the given id stretched to the width of w and h, in the position (x, y).

```
draw_surface_tiled(id, x, y)
```

Draws the surface with the given id tiled so that it fills the entire room. It starts the tile in the position (x, y).

```
draw_surface_part(id, left, top, width, height, x, y)
```

Draws the indicated part of the surface with the given id that is drawn at the position (x, y). The part of the surface starts and the position (left, top) with the width "width" and the height "height."

```
draw_surface_ext(id, x, y, xscale, yscale, rot, color, alpha)
```

Draws the surface with the given id in the position (x, y) scaled by xscale and yscale (1= no scaling) and rotated using the value rot (0=no rotation), and with blending color (use c_white for no blending) and the transparency alpha value from 0 to 1, whereas 0 is transparent completely.

```
draw_surface_stretched_ext(id, x, y, w, h, color, alpha)
```

Draws the surface with the given id stretched to the size of (w, h) in the position (x, y). Color is the blending color (c_white=no color blending) and alpha indicates the transparency setting from 0 to 1 whereas 0=transparent.

```
draw_surface_tiled_ext(id, x, y, xscale, yscale, color, alpha)
```

Draws the surface with the given id tiled so that it fills the entire room and starts in the position (x, y), but now with scale factors (xscale, yscale) and a color blending (color) and transparency setting (alpha) from 0 to 1 whereas 0 is transparent.

```
draw_surface_part_ext(id, left, top, width, height, x, y, xscale, yscale, color, alpha)
```

Draws the indicated part of the surface with the given id that starts on the position (left, top) of the surface, with

Surfaces Cont.

the width and height (width, height), with its origin at position (x, y) but now with scale factors (xscale, yscale) and a color blending (color) and transparency setting (alpha) from 0 to 1 whereas 0 is transparent.

```
draw_surface_general(id, left,
  top, width, height, x, y,
 xscale, yscale, rot, c1, c2,
       c3, c4, alpha)
```

The most general drawing function. It draws the indicated part of the surface with the given id, that starts on the position (left, top) in the surface, with the width and height of (width, height), with its origin at position (x, y) but now with scale factors (xscale, yscale), a rotation angle (rot), a color for each of the four vertices (top-left, top-right, bottom-right, and bottom-left) (c1,c2,c2,c4 respectively), and an alpha transparency value (alpha) from 0 to 1 whereas 0 is transparent.

Third Attempt

So in order to make the drawing on surfaces function work properly, you need to draw the surface in the draw event, all the other code that you have made in the create event is correct and must stay the same. Add the following code to the **Draw Event**.

```
draw_surface(surface,0,0); //Draws
the surface
```

Result

Now you can clearly see that the text is drawn, even though only the surface drawing function is called! Depending on your system and font-size settings, you may see that a part of the text is not drawn, that is because your surface width is "50," and there might not be room for all the text! This, ladies and gentlemen is the magic of surfaces!

Now let's move to the more-complex part of the surfaces, which is the once dealing with files and saving.

Secondary Code

Files and Saving

```
surface_save(id, fname)
```

This function saves the whole surface with the given id to a bitmap file. The filename must be a string (fname). The "fname" could be just the name of the file or its complete target.

```
surface_save_part(id, fname,
        x, y, w, h)
```

This function saves a part of the surface with the given id to a bitmap file. The filename must be a string (fname). The "fname" could be just the name of the file or its complete target. The part of the surface begins at the position (x, y) on the surface, and has the width (w) and height (h).

More Functions

```
surface_getpixel(id, x, y)
```

This function gets the color of the pixel in the position (x, y) in the surface with the given id.

```
surface_copy(destination, x,
        y, source)
```

This function copies the whole surface (source) to the new surface (destination). The (source) copied will be pasted in the position (x, y) of the new surface (destination). The copying occurs without doing any blending.

```
surface_copy_part(destination,
 x, y, source, xs, ys, ws, hs)
```

This function copies a part of the surface (source) to the new surface (destination). The (source) copied will be pasted in the position (x, y) of the new surface (destination). The part of the (source) to be copied will start on position (xs, ys) of the source and have the width (ws) and height (hs). The copying occurs without doing any blending.

Cleaner Surface Drawing

To draw to a cleaner surface, all add this piece of code after surface_set_target(id):

```
draw_clear_alpha(c_white,0);
```

This will make the surface completely transparent.

Special thanks & Closing Remarks

Thank you for reading this tutorial. The tutorial has been originally written by Eyas Sharaiha and published in the Game Maker community. It is now part of our wiki (GMking.org's Wiki), you can visit the wiki here:

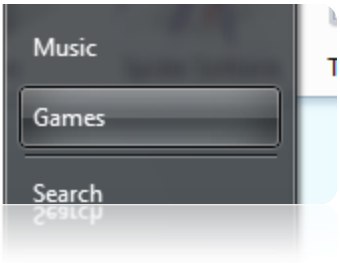
http://wiki.gmking.org/

Eyas Sharaiha

PROGRAMMING TUTORIAL

Integration with Vista's Game Explorer

A fabulous new feature in Microsoft's latest version of Microsoft Windows – Windows Vista – is the Game Explorer, and it is of particular interests to us game developers, and the gamers out there as well. This article will describe – in detail – how to integrate your own game with the Game Explorer.



Throughout the article, I will try to be as IDE-independent and language-independent as possible. This article will go through the whole process in great detail, so it'll be easy to understand both for novice developers and the "gurus".

The second section of this article, specifically integrating with the installer, is based off portions from the MSDN site at Microsoft. A more complete article will be released in the next issue of markup, showing how to integrate these objects with InstallShield itself.

Introduction of the Game Explorer

The Game Explorer is a new feature in Windows Vista that provides gamers with an excellent way to view information about the games they have, this includes:

1- The game title, description, version, release date, publisher, and developer information
2- Box-art image
3- Minimum and recommended system requirements
4- Game content rating and description

Other advantages the Game Explorer provides are:

- Ability of parental controls to control which game ratings could be played
- Unique settings for multiple installations on the same systems
- Customizable context menus displaying actions to be performed when right clicking a game

Adding your game!

In order to successfully create a game which could be added to the Game Explorer in Windows Vista, you must do two things:

1- Create certain settings and files that the system will read
2- Add these files and settings to the Installer of the game

Creating Data Files

There are specific types of data which will be read by the Game Explorer, you must create them in order to take advantage of the Game Explorer's capabilities.

Step 1: Create GDFs

A GDF is short for "Game Definition File" and it is an XML-based file carrying data about your game. In order for your information to be shown, you must create at least one of these GDF documents. If you want your game information to differ for each different world region, then you must create multiple GDFs, one for each region.

A Game Definition File has a certain format which you might abide by:

The Game Definition File Schema

There are certain data types allowed in Game Definition File (a table of which is at the bottom of the next page).

First of all, as with any XML document, you must define it:

```
<?xml version="1.0"
encoding="utf-16"?>
```

This defines the XML file as an XML 1.0 file, with a UTF-16 encoding. IF you do not know much about XML documents, then you can use the line above as it is, as UTF-16 is the most common encoding, and version 1.0 is the proper version to use.

Second, there's the **GameDefinitionFile** element. Now I've searched a lot about this element, but found very little explaining its syntax. I've seen the syntax of how a couple of games were written, and it seems to be the same, so it'd probably be okay to include this element as it is:

```
<GameDefinitionFile
xmlns="urn:schemas-microsoft-
com:GameDescription.v1"
xmlns:baseTypes="urn:schemas-
microsoft-
```

PROGRAMMING TUTORIAL

Vista's Game Explorer Cont.

```
com:GamesExplorerBaseTypes.v1">
```

Note that the GameDefinitionFile element will be wrapped around the rest of the XML document (the structure of the whole document will later be presented).

Inside the GameDefinitionFile, there's an element called the **GameDefinition** element. This contains two important attributes which need to be defined (see table at right).

GameDefination Element Attributes

Name	Data Type	Description
gameID	guid	The game's ID. This uniquely identifies your game title to the Game Explorer. This is generated by you, or, in the case of certain titles, has already been provided.
WMID	guid	The game's ID for Windows Metadata Services. This should only be used for legacy games.

An example is as follows:

```
<GameDefinition
gameID="GAMEID"
WMID="WMID">
```

An example includes:

```
<GameDefinition
gameID="{5C08D2FF-A48E-43ca-846C-
08924563E5A7}"
WMID="{5C08D2FF-A48E-43ca-846C-
08924563E5A7}">
```

Notice that both gameID and WMID are written in the format of a "guid" which is their data type. For each element, check its data type and make sure to write it in the proper criteria.

Please note that the GameDefinition element will also wrap around the rest of the document, so all later elements will be included in this element itself.

An important element is the **Name** element. Its data type is a boundedString, and it represents the name of the game.

```
<Name>Super Game 2005</Name>
```

There is an element called **description** with its data type as boundedString, which is an optional element to include.

```
<Description>The Penultimate
Game</Description>
```

Another optional element is the **ReleaseDate** element; its data type is "date" and is written as "yyyy-mm-dd":

```
<ReleaseDate>2004-06-
30</ReleaseDate>
```

The **version** element is a choice type which contains one of the two ways of specifying the version number of your game one method is an explicit version number embedded in the XML. The other method is a path to an executable or DLL where the version number can be extracted.

If it was to be written as a number, an element inside the main Version element will be added, called **VersionNumber** with fourPartVersion

Game Defination File Date Types

Name	Schema Component	Type	Description
fourPartVersion	simpleType	String (with facets)	A simple type that contains a regular expression for 4-part versions
guid	simpleType	String (with facets)	A simple type that contains a regular expression for guids. Guids must be of standard type, enclosed in braces.
boundedString	simpleType	String (with facets)	A string which preserves whitespace and is limited to 32768 characters.
filePathAttributeGroup	attributeGroup	N/A	Two frequently used attributes for file paths combined into an AttributeGroup.
filePathAttributeGroup: baseKnownFolderID	attribute	guid	A guid that represents a KnownFolder as the base path for folders. The path passed along is relative to this known folder. If this folder isn't specified then the path attribute below is relative to the game's installation directory.
filePathAttributeGroup: path	attribute	boundedString	A bounded string for the file path represented by this group. This file path is relative to the baseKnownFolderID above.

as its data type.

```
<Version>
  <VersionNumber
versionNumber="1.2.3.4"/>
 </Version>
```

If it was to be written as a path to a file which contains the version number inside it, an element inside the main Version element should be added, called **VersionFile** with filePathAttributeGroup as its data type.

```
<Version>
  <VersionFile
path="bin\MyGame.exe"/>
 </Version>
```

The **developers** element specifies the developers of this particular game, a link to their website can also be included:

```
<Developers>
  <Developer
URI="http://www.microsoft.com">Mi
crosoft</Developer>
 </Developers>
```

Note: A total of five different developer links could be added inside the developers element

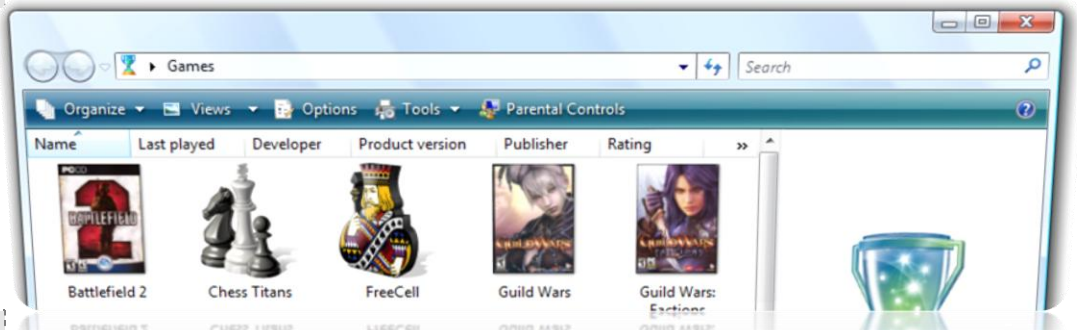
The **publishers** element specifies the publishers of this particular game, a link to their website can alos be included:

```
<Publishers>
  <Publisher
URI="http://www.microsoft.com">Mi
crosoft</Publisher>
 </Publishers>
```

Note: A total of five different developer links could be added inside the developers element

The **WindowsSystemPerformanceRating** element is an important but optional attribute that allows you to choose the recommended and required "Windows Experience Index" values for running your game.

Name	Data Type	Description
minimum	int	The minimum WSPR rating that a computer should have in order to play this game.
recommended	int	The recommended WSPR rating that a computer should have in order to play this game.




```
<WindowsSystemPerformanceRating
minimum="4" recommended="5"/>
```

The table below lists the attributes for WindowsSystemPerformanceRating.

There's also an element called the **genres** element, which could contain multiple **genre** elements, it looks as follows:

```
<Genres>
  <Genre>Action</Genre>
  <Genre>Adventure</Genre>
 </Genres>
```

Another element I really like is the **ratings** element which gives you multiple **rating** attributes for the game, so that parents or anyone else could know if the games suit themselves, their children, or any other person.

I'm not going to talk much about that element, but you can read more info at their MSDN article. Use the links at the end of the article.

The **Game Executables** element is the last one I'll talk about. It is very simple to use and understand:

```
          <GameExecutables>
           <GameExecutable
path="bin\game.exe"/>
          </GameExecutables>
```

An example of the complete code of the document is below:

Complete document code

```
<?xml version="1.0" encoding="utf-16"?>
<GameDefinitionFile xmlns="urn:schemas-microsoft-
com:GameDescription.v1"
xmlns:baseTypes="urn:schemas-microsoft-com:GamesExplorerBaseTypes.v1">
  <GameDefinition gameID="{dc90fdca-aa28-4d13-8401-ad149e4bccae}"
  WMID="{9e6c8124-5159-4aed-a175-a2dd292dfe86}">
    <Name>Boggle"!</Name>
    <Ratings>
     <Rating ratingID="{7a53b0be-b92d-4e8a-a11f-8e6f9f3c575b}"
      ratingSystemID="{768bd93d-63be-46a9-8994-0b53c4b5248f}" />
    </Ratings>
    <Version>
      <VersionNumber versionNumber="1.0.0.0" />
    </Version>
  </GameDefinition>
</GameDefinitionFile>
```

PROGRAMMING TUTORIAL

Vista's Game Explorer Cont.

Step 2: Create Bitmaps

The Game Explorer displays artwork for every game that provides it. The artwork provided is shown in a small format in the list of games, and in a large format when a game is selected by the user. Create an image file that will represent your game in the Game Explorer.

The image should be in PNG format, as this allows for transparency features to give your artwork a clean look in the Game Explorer.

Step 3: Embed Bitmaps and GDFs

After creating the GDF and Bitmap files, you can now embed them all together. The GDF files and bitmap files used by Game Explorer cannot be independent files that are simply installed into your game directory along with the rest of the game files. They must be embedded as resources, either into an executable file, or a DLL.

The Game Explorer header (GameUx.h) contains defined constants that you must use in the resource script to identify your GDF file and bitmap file so that the Game Explorer can find them. By separating the resource script into #ifdef sections for multiple languages, you can instruct Game Explorer to search for localized versions of GDF files and bitmaps.

Adding the Game from the Installer

Step 1: Install the game files

The first step in adding your game to the Game Explorer is to have your game files installed on the hard drive. It is important that all game files be installed by this step; once the game is added to the Game Explorer in step 4, Parental Controls may prevent your install process from accessing the game's base install directory to add or change files.

Step 2: Create an IGameExplorer COM Object in the Installation

Your next step would be creating an IGameExplorer COM object. As seen in reference (4).

Step 3: Call function "VerifyAccess" in the IGameExplorer

The function VerifyAccess must be called from the IGameExplorer COM object. It will check if the game has permission to run on the current user account. As seen in reference (5).

Step 4: Call function "AddGame" in the IGameExplorer

The function AddGame must be called from the IGameExplorer COM object. You must call AddGame with the path to your GDF resource, base install path, and install scope, and optionally an InstanceID GUID.

The add game function has the following syntax:

```
HRESULT AddGame(
  const BSTR bstrGDFBinaryPath,
  const BSTR
bstrGameInstallDirectory,
  GAME_INSTALL_SCOPE
installScope,
  GUID* pguidInstanceID
);
```

For more information, see ref. (6).

Step 5: Persist Game InstanceID

You must persist the InstanceID that is returned from the AddGame function. This is needed in order to be able to properly uninstall the game.

Step 6: Create Game Task Folders

As seen in reference (9), you must use the InstanceID to create a subdirectory of that ID either in the common task directory used for all files or in the user-specific task directory. In that folder, you should create the tasks that could be done with the game.

Eyas Sharaiha

References

1. **Main Page**, http://msdn2.microsoft.com/en-us/library/ms687240.aspx
2. **Defined constants of GameUx.h,** http://msdn2.microsoft.com/en-us/library/ms687210.aspx
3. **GDF file schema,** http://msdn2.microsoft.com/en-us/library/ms687239.aspx
4. **IGameExplorer COM**, http://msdn2.microsoft.com/en-us/library/ms687221.aspx
5. **Verify Access**, http://msdn2.microsoft.com/en-us/library/ms687229.aspx
6. **AddGame,** http://msdn2.microsoft.com/en-us/library/ms687225.aspx
7. **Game Explorer Tasks,** http://msdn2.microsoft.com/en-us/library/ms687246.aspx
8. **Registry information (XP)**, http://msdn2.microsoft.com/en-us/library/ms687245.aspx
9. **Create Game Task Folders,** http://msdn2.microsoft.com/en-us/library/ms687246.aspx

PROGRAMMING TUTORIAL

C++ “Hello World” tutorial

C++ is one of the most popular programming languages out there, but learning can be a tedious process.

One of the first things anyone learning C++ should know is outputting text. Unlike more recent software, no function exists in C++ “as is” that could display text; instead, you need include “libraries” that would introduce such functions and capabilities.

To start the tutorial, open a new C++ project, and start working on the main .cpp file. (Need a development tool? Try Microsoft Visual C++ 2005 Express Edition SP1, you can get it from http://msdn.microsoft.com/vstudio/express and, it’s totally free!)

The Code

The very first thing you should include is Input/output stream that would allow you to output text. In older versions of C++, you could include a header file (.h file) that would give you these functions, but now it is done differently:

```
#include <iostream>
```

To start giving out instructions to output the text, you should first initialize the main function that would execute the actual code. You could do so by typing:

```
int main() {
…
}
```

The "int" stands for integer. That means "int" initializes the function main()’s returned value as an **integer**. This might be hard to understand when you first start programming C++, but that's normal. The important thing is knowing that the line of code "int main()" initializes the code. It is in this function that all the actions occur.

Many also like to use:

```
void main()
```

But I personally – for some weird reason – prefer using int, so that’s how we’re going to continue.

The curled brackets you saw earlier define the beginning and ending of a block of code. All the code that we will perform in this tutorial must be in the int main() block of code.

The actual code to output the “Hello world!” text is:

```
std::cout << "Hello World!!\n";
```

The “std::cout<<” is the command that tells the computer to output text. The text written later must be a “string” – so text and numbers should be surrounded by quotation marks.

The final “\n” you see at the end defines a new line, similar to the Enter Key we use when typing text in notepad or word. If this wasn’t use, the “Insert any key to continue” text that is normally automatically outputted by C++ would be right next to the “!!” in the text you outputted.

Another important thing is that all instructions **must** end with a semicolon (;), otherwise an error is outputted.

Finally, to conclude the function “main”, we need to write a return instruction, this might seem pretty meaningless now, but most functions return values of some sort, so the return itself is used to mark the end of a function.

Since we really do not need to return any value, let’s just choose to return “0”:

```
return 0;
```

Final Code Form

```
#include <iostream>
int main() {
  std::cout << "Hello World!!\n";
  return 0;
}
```

Running the code

You can build the C++ project or you can use the Debug function to run it. Note: in Visual C++ 2005 Express Edition, you will not get a "Press Any Key to Continue . . ." message, and the program ends instantly if it is run in debug mode, therefore you will see no result. To fix that, click on the Debug menu on top, and then click on the menu item "Start Without Debugging".

End Result

When running the program, you should see the following:

```
Hello World!!
Press any key to continue . . .
```

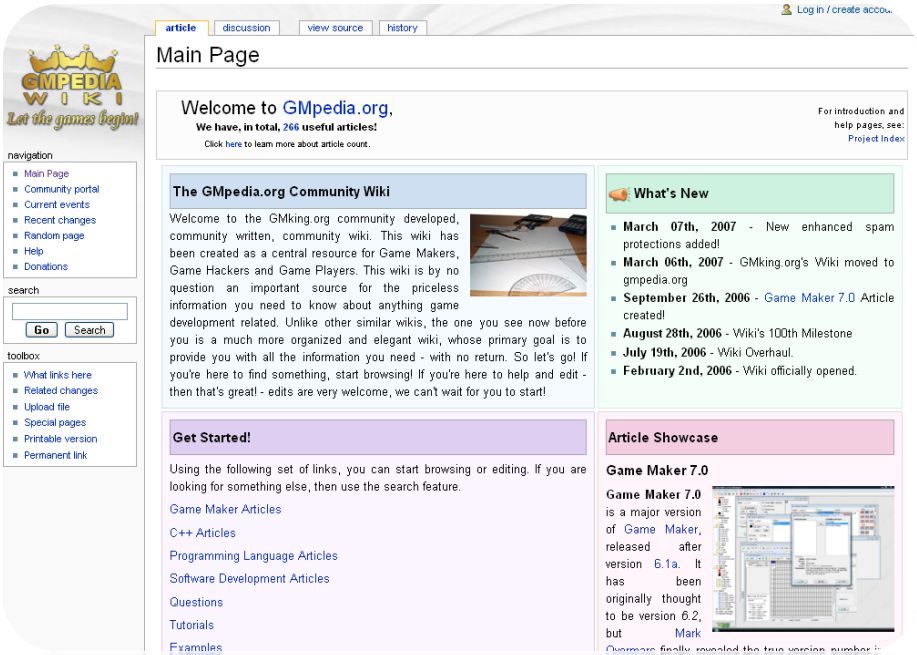
However, if you have omitted the “\n” (newline) at the end of the code, as stated earlier, you would see:

```
Hello World!!Press any key to
continue . . .
```

Eyas Sharaiha

That's all for this issue!

... but check our other online resources in the meantime...



gmpedia.org

forums.gmking.org

openload.info

ircbloopers.com

gmking.org

irc://irc.gamemaking.org/#gmking

GMking.org Let them make games!